```
// FILE: xl320-line1_0.nxc
// DATE: 01/22/20 17:12
// AUTH: P.Oh
// DESC: XL-320 based 2-DOF planar manipulator.  Trajectory planned based on
//       a parametric line.  User prescribes the desired number of equally
//       spaced points given start and end points of that line
// VERS: 1.0a: xl320-twoLinkFunctions1_0a.h contains functions
//       rotateMotorAbsolutely(); goHome(); twoLinkInverseKinematics()
// REFS: H-files xl320-defines1_0a.h and xl320-functions1_0d.h
// NOTE: If factory default XL-320 used, then ID is 0x01
//       ID of 0xFE commands any and all XL-320 motors
//       This example uses an XL-320 configured with ID# 3

#include "xl320-defines1_0a.h"
#include "xl320-functions1_0d.h"
#include "xl320-twoLinkFunctions1_0a.h"

// Global variables
bool  orangeButtonPushed;       // Detect Brick Center button state
bool  rightArrowButtonPushed;   // Detect Brick right arrow button state

task main() {

  // planar manipulator variables
  float l1, l2; // length of link 1 and link 2 [mm]

  float xDesired[], yDesired[]; // desired line's (x,y) way points
  int numberOfWayPoints;        // # of points between start and end points
  int numberOfSpaces;           // # of equally spaced segments
  int maxVectorSize;            // # array elements = numberOfWayPoints + 2
  float xP, yP;                 // EE absolute position wrt x0y0 frame [mm]
  int i, j;                     // dummy index variables
  string str01, str02;          // dummy string variables to display text
  float t;                      // variable for parametric equation of line

  THETA_IN_DEGREES anglesInDegrees;  // struct defined in
  THETA angles;                      // xl320-twoLinkFunctions1_0a.h

  // Initializations
  l1 = 7 * mmPerStud;            // [mm] link 1 is 7 studs long
  l2 = 5 * mmPerStud;            // [mm] link 2 is 5 studs long

  // Define and initialize arrays that will hold waypoints
  numberOfWayPoints = 14;                  // start stud + (7 studs) + end stud
  numberOfSpaces = numberOfWayPoints + 1; // i.e. hence 8 equally spaced segments
  maxVectorSize = numberOfWayPoints + 2;  // include both start and end points
  ArrayInit(xDesired, 0, maxVectorSize);  // initialize waypoint x vector to 0
  ArrayInit(yDesired, 0, maxVectorSize);  // initialize waypoint x vector to 0
  // Initialize start and end points of line
  xDesired[0] = 6*mmPerStud;                    // [mm] starting point
  yDesired[0] = 8*mmPerStud;                    // [mm] starting point
  xDesired[numberOfWayPoints+1] = 6*mmPerStud;  // [mm] ending point
  yDesired[numberOfWayPoints+1] = -8*mmPerStud; // [mm] ending point

  // Parametric equation of line to calculate equally spaced points
  i = 1;
  while (i <= (numberOfWayPoints)) {
    t = i/(numberOfWayPoints+1);
    xDesired[i] = 6*mmPerStud; // [mm]
    yDesired[i] = (yDesired[0]*(1.0-t)) + (t*yDesired[numberOfWayPoints+1]);
    i++;
  } // end of while
  sprintf(str01, "t=%3.3f " , 1.0/(numberOfWayPoints+1) );
  sprintf(str02, " N=%d" , numberOfWayPoints);
  TextOut(0, LCD_LINE1, strcat(str01, str02));
  TextOut(0, LCD_LINE2, FormatNum("yD[0] = %3.2f mm" , yDesired[0]));
  sprintf(str01, "yD[%d]" , numberOfWayPoints + 1);
  sprintf(str02, "%3.2f mm" , yDesired[numberOfWayPoints + 1]);
  TextOut(0, LCD_LINE3, strcat(str01, str02) );
  TextOut(0, LCD_LINE5, "Cont'd ORG" );

  do {
    orangeButtonPushed = ButtonPressed(BTNCENTER, FALSE);
  } while(!orangeButtonPushed);

  UseRS485();
  RS485Enable();
  RS485Uart(HS_BAUD_57600, HS_MODE_8N1); //57600 baud, 8bit, 1stop, no parity

  // Prompt user to begin
  ClearScreen();
  TextOut(0, LCD_LINE1, "Start: hit ->");
  do {
     rightArrowButtonPushed = ButtonPressed(BTNRIGHT, FALSE);
  } while(!rightArrowButtonPushed);
  ClearScreen();

  // (1) go to home position
  TextOut(0, LCD_LINE1, "Homing..." );
  Wait(2000);
  goHome();
  Wait(2000);
  TextOut(0, LCD_LINE7, "Homed" );
  PlayTone(TONE_E4, 500);
  TextOut(0, LCD_LINE8, "Cont'd ORG" );
  do {
    orangeButtonPushed = ButtonPressed(BTNCENTER, FALSE);
  } while(!orangeButtonPushed);

  // (2) move to start of line
  ClearScreen();
  TextOut(0, LCD_LINE1, "Going (xP0,yP0):" );
  Wait(2000);
  xP = xDesired[0];
  yP = yDesired[0];
  sprintf(str01, "(%3.2f," , xP/mmPerStud);
  sprintf(str02, "%3.2f) stud" , yP/mmPerStud);
  TextOut(0, LCD_LINE2, strcat(str01, str02));
  anglesInDegrees = twoLinkInverseKinematics(l1, l2, xP, yP);
  rotateMotorAbsolutely(anglesInDegrees.theta1InDegrees,
anglesInDegrees.theta2InDegrees);
  TextOut(0, LCD_LINE7, "Now @ line start" );
  TextOut(0, LCD_LINE8, "Start: hit ->");
  do {
     rightArrowButtonPushed = ButtonPressed(BTNRIGHT, FALSE);
  } while(!rightArrowButtonPushed);

  // (3) Iterate thru waypoints, calculate IK, and command motor angles
  for(i=1; i <= numberOfWayPoints+1; i++) {
      xP = xDesired[i];
      yP = yDesired[i];
      anglesInDegrees = twoLinkInverseKinematics(l1, l2, xP, yP);
      sprintf(str01, "(%3.2f," , xP/mmPerStud);
      sprintf(str02, "%3.2f) stud" , yP/mmPerStud);
      TextOut(0, LCD_LINE2, strcat(str01, str02));
      // Actuate the XL-320 motors
      rotateMotorAbsolutely(anglesInDegrees.theta1InDegrees, 
anglesInDegrees.theta2InDegrees);
  }; // end for-loop

  // (4) Lastly, since trajectory done, go home
  Wait(3000);
  TextOut(0, LCD_LINE7, "Line done" );
  TextOut(0, LCD_LINE8, "Go Home: Hit ORG" );
  do {
    orangeButtonPushed = ButtonPressed(BTNCENTER, FALSE);
  } while(!orangeButtonPushed);
  ClearScreen();
  goHome();
  TextOut(0, LCD_LINE4, "Homed. Exiting" );
  Wait(3000);
  PlaySound(SOUND_DOUBLE_BEEP);
} // end main
```